Supervisory Control of Product and Hierarchical Discrete Event Systems

Benoit Gaudin, Hervé Marchand

Supervisory Control of Product and Hierarchical Discrete Event Systems

B. Gaudin, H. Marchand,
VerTeCs Team, Irisa,
Campus Universitaire de Beaulieu, 35042 Rennes, France,
E-mail: {bgaudin,hmarchan}@irisa.fr, Fax: +33 2 99 84 71 71

November 5, 2003

Abstract

In this paper, the supervisory control of a class of Discrete Event Systems is investigated. Discrete event systems are modeled either by a collection of Finite State Machines that behave asynchronously or by a Hierarchical Finite State Machine. The basic problem of interest is to ensure the invariance of a set of particular configurations in the system. When the system is modeled as asynchronous FSMs, we provide algorithms that, based on a particular decomposition of the set of forbidden configurations, solve the control problem locally (i.e. on each component without computing the whole system) and produce a global supervisor ensuring the desired property. We then provide sufficient conditions under which the obtained controlled system is non-blocking. This kind of objectives may be useful to perform dynamic interactions between different parts of a system. Finally, we apply these results to the case of Hierarchical Finite State Machines.

Keywords: Supervisory Control Theory, structured FSM, structured Supervisor, modularity and non-blocking.

1 Introduction

In this paper we are interested in the Supervisory Control [17] of structured Discrete Event Systems (DES). The system to be controlled is modeled as a collection of Finite State machines (FSM) that behave asynchronously or by Hierarchical Finite Machine (by adding nested FSMs). In many applications and control problems, FSMs are the starting point to model fragments of a large system, which usually consists of the composition and of the nesting of many different sub-systems. The standard way of applying the supervisor synthesis methodology to such systems is by expanding them to ordinary state machines and by using classical synthesis tools on the resulting FSM. However, knowing that the synthesis algorithms are polynomial in the number of states of the systems and that the number of states of the global systems grows exponentially with the number of parallel and nested sub-systems, it is important to design algorithms that perform the controller synthesis phase by taking advantage of the structure of the system without expanding it. In other words, given the modular structure of the system, it becomes of interest, for computational reasons, to be able to synthesize a supervisor on each sub-part of the system and then to infer a global supervisor from the local ones.

Several approaches have been considered in the literature to deal with reducing the complexity of supervisor synthesis. Modular control [23] and modular plant [6, 7] are natural ways to handle this problem. In [23], the method consists in dividing the global control objectives into sub-objectives and to perform the controller synthesis phase w.r.t. these sub-objectives. In [6, 7], the authors considers product systems (i.e. systems

composed of asynchronous subsystems, not sharing common events). Given a control objective, a local system is built from the sub-systems that are coordinated by the control objective (i.e. all the sub-systems that share some events used to express it). It is then sufficient to compute the local supervisor ensuring the control objective with respect to this local system in order to obtain the result on the whole system. Finally, the modularity property [23] ensures that a maximal solution of the control problem can be obtained by computing the local supervisors w.r.t. to the local control objectives[1]. A similar work based on an incremental synthesis methodology can be found in [1]. The fact that only a sub-part of the whole system has to be built in order to compute a supervisor obviously decreases the complexity of its computation. However, it may happen that given a control objective, the whole plant need to be computed (e.g. if all the events of the alphabet of the plant are necessary to express the objective). In [11], the authors provide an elegant way to solve the SCP when both the plant and the specification (or control objective) are given in a modular way. The controller synthesis is then performed by computing supervisors that are further added to the specification, itself seen as the initial supervisor). Note that in the above approach, no particular connection between the components of the plant and of the specification are required. In contrast, we adopt a state-based approach in which the control objective exactly fits with the structure of the plant.

In order to take into account nested behaviors, some techniques based on state-based or language based aggregation methods [22, 4, 19] have been proposed to deal with hierarchical control problems. However, in this paper, we are more interested in applying existing techniques to a multi-level hierarchy model rather than in abstracting the original system using a bottom-up approach. In other words, the system is already specified in a hierarchical way and we want to perform control on it without changing its structure. The notion of Hierarchical state machines was first popularized by Harel in [10], who introduced the STATECHARTS model. STATECHARTS are a very rich graphical specification formalism allowing (among others) the nesting of state machines (inducing an implicit hierarchy), the orthogonality of state machines (inducing parallelism between state machines) and re-usability of components in different contexts. For supervisory controller purposes, Brave and Heimann in [2] introduced Hierarchical State Machines which constitute a simplified version of the STATECHARTS. Compared to the classical state machines, they add orthogonality and hierarchy features. In this approach, even if computations are locally performed, the obtained supervisor does not reflect the structure of the system. Some other works dealing with control and hierarchy can be found in [9, 12, 13]. In [9], the system is described by means of a hierarchical language, but at the low level of the hierarchy, there is no notion of interaction between components (i.e. the orthogonality feature is not taken into account). In [12, 13], the interaction between the levels of the systems are defined by *interfaces*. Now, given a set of supervisors (one for each component of the system), they give conditions under which the *controlled system* is controllable and non-blocking. However, no algorithm is presented to compute this set of supervisors according to some control objective.

In the first part of this paper, we focus on product systems (i.e. the whole systems is modeled as a collection of plants $(G_i)_{i\leq n}$ that behave asynchronously) and we present a methodology that solves the Supervisory Control Problem for a particular class of control objectives that fit with the structure of the plant. More precisely, the problem of interest is to ensure the invariance of a set of configurations in the global system. Based on a particular decomposition of this set in terms of set product, we provide algorithms that locally solve the control problem (i.e. on each component without computing the whole system) and produce a global supervisor ensuring the desired property. This supervisor can be seen as an oracle that will activate/deactivate local supervisors according to the current configuration of the global system and some conditions that can be easily computed on the fly. Moreover, we make the necessary efforts to keep the structure of the plant in the global supervisor, hence improving the readability and the understanding of the supervisor effect as well as its memory storage. One example of control for such systems would be to avoid a press to be in the closed position when the arm of a robot is located inside the press. The plant is then

[1]Note that the authors also give necessary and sufficient conditions under which the obtained controlled system is non-blocking

composed of two sub-systems: a press and an articulated arm, that has to place (to remove) a plate inside (from) the press. One can see that global states such as "*the press is closed*" while "*the arm is located inside in the press*" have to be avoided during the execution of the system. Separately, they do not correspond to a dangerous state. It is only when all sub-systems are simultaneously in these particular states that the system is itself in a dangerous situation that has to be avoided. The intuitive idea of our method is then the following: we first compute a supervisor acting upon the press subsystem, for which the goal is to avoid the press to be closed and a supervisor acting upon the arm subsystem, for which the goal is to avoid the arm to be inside the press. Finally, we combine the two supervisors in order to achieve the global specification.
In the second part, we extend these notions to the the case of Hierarchical Finite State Machines (HFSM). The HFSM we are considering can be characterized by a collection of nested structures $\langle K_1, \ldots, K_n \rangle$, where K_1 represents the top-level of the HFSM. At an intermediate level, the structure K_i can be seen as an HFSM, in which states can be either ordinary states or super-states b which are constituted by a set of structures $(K_j)_{j \in J_b} \subseteq 2^{\langle K_{i+1}, \ldots, K_n \rangle}$, running in parallel. Each structure could have several initial states and a unique final state. The informal behavior is the following: whenever the system evolves into a super-state b, all the structures of this super-states are simultaneously activated and entered in one of their initial states, depending on the event that has been triggered to enter the super-state b. *A contrario*, it is possible to evolve out a super-state b, whenever all the structures of this super-state are in their corresponding final-state (i.e., there is no preemption; the output of a super-state is synchronized with the end of each of the tasks associated with the different structures involved in this super-state). Inside a super-state, the different structures evolve asynchronously. Based on this model, we define algorithms that solve the State Avoidance Control Problem for configurations that can be expressed as a Cartesian product of local forbidden state sets.
Note that for both models (modular and hierarchical), the controlled system has the same behavior as the one that would have been obtained by first expanding the system and by applying the classical algorithms on the resulting FSM.
The remainder of this paper is organized as follows. Section 2 consists of a presentation of the basic model on which control will be applied as well as a review of the classical controller synthesis methodology [17]. In section 3, we propose a modular methodology to solve the State avoidance Control problem for a plant modeled as a collection of asynchronous sub-plants. The supervisor described in Section 3.2 could be blocking. Then, in section 3.3, we provide sufficient conditions under which the obtained controlled system is non-blocking. In Section 4, after a brief presentation of the HFSM, we extend the results of Section 3 to this new model. Finally, Section 5 presents conclusions, and future work.

2 Preliminaries

In this section, the main concepts and notations are defined. More definitions will be given in the following sections. The reader is referred to [5] for any undefined concept.

2.1 The basic model.

The basic structures from which the plant will be built are Finite State Machines (FSMs) [5], that are defined as follows:

Definition 1 *A Finite State Machine (FSM) is defined by a 5-tuple $G = \langle \Sigma, \mathcal{X}, \mathcal{X}_o, \mathcal{X}_f, \delta \rangle$, where Σ is the finite alphabet of G. $\mathcal{X}$ is the finite set of states, $\mathcal{X}_o \subseteq \mathcal{X}$ is the set of initial states[2], whereas $\mathcal{X}_f$ is the set of final (marked) states of G, δ is the partial transition function defined over $\Sigma \times \mathcal{X} \longrightarrow \mathcal{X}$.*

[2]The fact that we consider FSMs with multiple initial states will be useful in Section 4 for the Hierarchical Finite State Machines.

We can think of G as an uncontrolled plant that starts at $x_o \in \mathcal{X}_o$ and executes/generates a sequence of events that are accepted by δ. The notation $\delta(\sigma, x)!$ means that $\delta(\sigma, x)$ is defined, i.e., there is a transition labeled by an event σ out of state x in machine G. Likewise, $\delta(s, x)$ denotes the state reached by taking the sequence of events defined by trace s from state x in machine G. $\delta(x)$ denotes the active event set of x. Similarly, $\delta^{-1}(x)$ denotes the set of events that lead to x. The behavior of the system is described by the language generated by G (i.e. $\mathcal{L}(G) = \{s \in \Sigma^* \mid \exists x_o \in \mathcal{X}_o,\ \delta(s, x_o)!\}$).

Blocking. An FSM is said to be *blocking* if

- either it exists a reachable state x, such that $\delta(x) = \emptyset$ but $x \notin \mathcal{X}_f$. This is called a *deadlock* because no event can be triggered.
- Or, there is a set of unmarked states in G that forms a strongly connected component, but with *no transition going out of the set*. If the plant enters this set of states, then we get what is called a *livelock*.

In the sequel, we assume that G is *trim* with respect to $\mathcal{X}_o$ and $\mathcal{X}_f$. This assumption means that all the states of G are accessible from one of the initial state x_0 and coaccessible to the marked states $\mathcal{X}_f$ (i.e. $\forall x \in G$, there exists $s, t \in \Sigma^*, x_o \in \mathcal{X}_o$ and $x_f \in \mathcal{X}_f$, st, $\delta(s, x_o) = x$ and $\delta(t, x) = x_f$). If G is trim then it can be shown that G is also non-blocking.

Submachines. We now introduce the notion of submachines of an FSM [5]. This notion will be useful for control purposes.

Definition 2 *An FSM $H = \langle \Sigma_H, \mathcal{X}_H, \mathcal{X}_{H_o}, \mathcal{X}_{H_f}, \delta_H \rangle$ is a submachine of G, denoted $H \subseteq G$, if $\Sigma_H = \Sigma,\ \mathcal{X}_H \subseteq \mathcal{X},\ \mathcal{X}_{H_o} \subseteq \mathcal{X}_o,\ \mathcal{X}_{H_f} \subseteq \mathcal{X}_f,\ \forall \sigma \in \Sigma_H, \forall x \in \mathcal{X}_H\ \ \delta_H(\sigma, x)! \Rightarrow (\delta_H(\sigma, x) = \delta(\sigma, x))$.*

2.2 Review of the Supervisory Control Problem

Supervisory control theory deals with control of Discrete event systems. The idea of this theory is that the plant models the uncontrolled behavior, which is not fully satisfactory. Hence, its behavior has to be modified by means of a feedback control (named Supervisor) in order to achieve a given set of requirements that the initial DES did not satisfy [17]. In practice, one of the main control problem is the invariance problem (or dually the state avoidance control problem), i.e. the supervisor has to control the plant so that the controlled plant remains in a safe set of states, or dually do not reach a set of forbidden states. In the literature, this invariance problem is often expressed using predicates over the states of the plant [16, 20, 24]. The control problem is then to force the system to remain in the states that satisfy the predicates.
Assume a plant G is given and modeled as an FSM and a set of states E, we recall how to synthesize a supervisor that will ensure the avoidance of a given set of states E. Knowing that some events are uncontrollable Σ_{uc}, as opposed to the set of controllable events (Σ_c), we first recall the definition of a *controllable submachine*.

Definition 3 *Let G be a FSM and H be a submachine of G, then H is controllable w.r.t. G and Σ_{uc}, whenever $\forall x \in \mathcal{X}_H \subseteq \mathcal{X}, \forall \sigma \in \Sigma_{uc}, \delta(\sigma, x)! \Rightarrow \delta_H(\sigma, x)!$.*

A supervisor $\mathcal{S} = (S, \mathcal{X}'_o)$ is given by a function $S : \mathcal{X} \to 2^{\Sigma}$, delivering the set of actions that are disabled in state x of G by control, and the new set of valid initial states $\mathcal{X}'_o \subseteq \mathcal{X}_o$ (it could be the case, that in order to ensure an objective, we need to reduce $\mathcal{X}_o$).

Remark 1 *In a more general framework, S is a function from $\mathcal{L}(G)$ into 2^{Σ}. However, for the control objectives we have in mind (ensuring the invariance/avoidance of a set of states), memory-less supervisors are sufficient since the resulting controlled system happens to be a sub-machine of the original plant.* $\diamond$

We write $\mathcal{S}/G$ the system, consisting of the initial plant G controlled by the supervisor $\mathcal{S}$. The closed-loop system $\mathcal{S}/G$ is actually a submachine of the initial plant such that the transition relation δ_s of $\mathcal{S}/G$ is obtained by restricting δ according to the control policy of the supervisor, i.e.

$$\forall x, x' \in \mathcal{X},\ \forall \sigma \in \Sigma,\ \delta_s(\sigma, x) = x' \Leftrightarrow \delta(\sigma, x) = x' \ \wedge \sigma \notin S(x)$$

and by keeping only the accessible part of this submachine.

The State Avoidance Control Problem (SACP) *given G and E a set of states, the problem is to build a supervisor $\mathcal{S}$ such that (1) $\mathcal{S}/G$ is controllable w.r.t. G and Σ_{uc}, (2) the traversed states do not belong to E and (3) $\mathcal{S}/G$ is the most permissive solution, i.e. for all other supervisor $\mathcal{S}'$ satisfying (1) & (2), $\mathcal{S}'/G$ is a submachine of $\mathcal{S}/G$ (i.e. $\mathcal{S}'/G \subseteq \mathcal{S}/G$). Such a supervisor is said to be maximal.*

In order to compute such a supervisor, we classically introduce two sets of states: the *weak forbidden set* of states and the *border set* that will be intensively used in the remainder of this paper.

Definition 4 *Given an FSM $G = \langle \Sigma, \mathcal{X}, \mathcal{X}_o, \mathcal{X}_f, \delta \rangle$, and a set of states $E \subseteq \mathcal{X}$, we denote by $\mathcal{I}(E)$ and $\mathcal{F}(E)$ the weak forbidden states and the border set of E, that are formally defined by:*

$$\mathcal{I}(E) = \{x \in \mathcal{X} \mid \exists s \in \Sigma_{uc}^*, \delta(s, x) \in E\} \tag{1}$$

$$\mathcal{F}(E) = \{x \in \mathcal{X} \setminus \mathcal{I}(E) \mid \exists \sigma \in \Sigma, s.t.\ \delta(\sigma, x) \in \mathcal{I}(E)\} \tag{2}$$

$\mathcal{I}(E)$ corresponds to the set of states from which it is possible to evolve into E by a trace of uncontrollable events, whereas $\mathcal{F}(E)$ corresponds to the set of states from which it is still possible to perform a control on G before evolving into $\mathcal{I}(E)$. The next proposition is adpated from [16].

Proposition 1 *Given an FSM G and $E \subseteq \mathcal{X}$, a set of states to be forbidden by control, the supervisor $\mathcal{S}$ of G given by the pair $(S, \mathcal{X}'_o)$, such that*

$$\begin{array}{rcl} \forall x \in \mathcal{X},\ S(x) & = & \begin{cases} \{\sigma \in \Sigma_c \mid \delta(\sigma, x) \in \mathcal{I}(E)\} \text{ if } x \in \mathcal{F}(E) \\ \emptyset \text{ Otherwise} \end{cases} \\ \mathcal{X}'_o & = & \mathcal{X}_o \setminus \mathcal{I}(E) \end{array} \tag{3}$$

ensures the avoidance of E in G and is maximal. $\diamond$

If $\mathcal{X}'_o = \emptyset$, then it means that the BSACP has no solution. In this case, the obtained supervisor $\mathcal{S} = (S, \emptyset)$ will be called the *trivial supervisor*. This notion will be useful in the next section.

Example 1 *Let us consider, the following FSM, where the state x_3 is a forbidden state, $\Sigma_c = \{a, b\}$ and $\Sigma_{uc} = \{uc\}$. The weak forbidden set of states $\mathcal{I}(\{x3\}) = \{x_3, x_1\}$, whereas the border of $\{x_3\}$ is given by $\mathcal{F}(\{x_3\}) = \{x_0, x_2\}$.*

The resulting Supervisor is given by $\mathcal{S} = (S, \{x_o\})$, where S si defined by: $S(x_o) = S(x_2) = \{a\}$ and for all the other states x, $S(x) = \emptyset$.

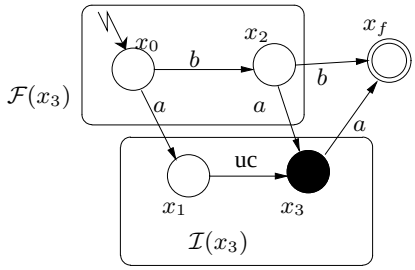


Figure 1: The BASCP: a small Example

Proposition 2 *[Modularity] let G be a plant and E_1, E_2 be two set of forbidden states. Let $\mathcal{S}_1 = (S_1, \mathcal{X}_{o_1})$ (resp. $\mathcal{S}_2 = (S_2, \mathcal{X}_{o_2})$) be the maximal supervisor ensuring the avoidance of E_1 (resp. E_2), then $\mathcal{S} = (S, \mathcal{X}'_o)$, where $\forall x \in \mathcal{X}$*

$$\begin{cases} S(x) = S_1(x) \cup S_2(x) \\ \mathcal{X}'_o = \mathcal{X}_{o_1} \cap \mathcal{X}_{o_2} \end{cases} \tag{4}$$

ensures the avoidance of $E_1 \cup E_2$ and is maximal.

Results is due to [16] and basically comes from the fact that $\mathcal{I}(E_1 \cup E_2) = \mathcal{I}(E_1) \cup \mathcal{I}(E_2)$ and $\mathcal{F}(E_1 \cup E_2) = (\mathcal{F}(E_1) \cup \mathcal{F}(E_2)) \setminus (\mathcal{I}(E_1 \cup E_2))$.

3 Control of Product Systems

In this section we are interested in a plant G modeled as a set of FSMs $(G_i)_{i \leq n}$, that behave asynchronously[3]. After a presentation of the model, we first present the state avoidance control problem for such plants, and then give a sufficient condition under which the obtained supervisor is non-blocking.

3.1 The Model

The plant G, we now consider is modeled as a collection of FSMs $(G_i)_{i \leq n}$, that behave asynchronously. Such plants are called product plants in [6]. G is then given by the FSM $G_1 \parallel \cdots \parallel G_n$, where the operation $\parallel$ is defined by the following definition:

Definition 5 *Consider two FSMs, $G_1 = \langle \Sigma_1, \mathcal{X}_1, \mathcal{X}_{o_1}, \mathcal{X}_{f_1}, \delta_1 \rangle$ and $G_2 = \langle \Sigma_2, \mathcal{X}_2, \mathcal{X}_{o_2}, \mathcal{X}_{f_2}, \delta_2 \rangle$, with $\Sigma_1 \cap \Sigma_2 = \emptyset$. The asynchronous product of G_1 and G_2, noted $G_1 \| G_2$, is another FSM $\langle \Sigma_{12}, \mathcal{X}_{12}, \mathcal{X}_{o_{12}}, \mathcal{X}_{f_{12}}, \delta_{12} \rangle$, such that $\Sigma_{12} = \Sigma_1 \cup \Sigma_2$, $\mathcal{X}_{12} = \mathcal{X}_1 \times \mathcal{X}_2$, the new set of initial states is given by $\mathcal{X}_{o_{12}} = \mathcal{X}_{o_1} \times \mathcal{X}_{o_2}$ and the set of final states by $\mathcal{X}_{f_{12}} = \mathcal{X}_{f_1} \times \mathcal{X}_{f_2}$. The partial transition function δ_{12} is defined by:*

$$\delta_{12}(\sigma, \langle x_1, x_2 \rangle) = \begin{cases} \langle \delta_1(\sigma, x_1), x_2 \rangle \text{ if } \sigma \in \Sigma_1 \text{ and } \delta_1(\sigma, x_1)! \\ \langle x_1, \delta_2(\sigma, x_2) \rangle \text{ if } \sigma \in \Sigma_2 \text{ and } \delta_2(\sigma, x_2)! \\ \text{Undefined otherwise} \end{cases} \tag{5}$$

Throughout the remainder of this section, the different components of the plant are designed to be non-blocking (i.e. each FSM G_i is assumed to be trim). This clearly entails that $G = G_1 \parallel \cdots \parallel G_n$ is also non-blocking.

[3] As example of such a plant, one can see the modeling and control of an excavator machine that can be found in [18].

Notations. Following Definition 5, states of the plant G will have the form $\langle x_1, \ldots, x_n \rangle$, where each $x_i \in \mathcal{X}_i$ of G_i. For convenience, we will call a *state*, any element $x_i \in \mathcal{X}_i$ of a particular FSM G_i, and a *global state* a tuple of the form $\langle x_1, \ldots, x_n \rangle$ of G, i.e. a "state" of the resulting FSM. Moreover we call $\mathcal{X}^F = \mathcal{X}_1 \times \cdots \times \mathcal{X}_n$ the set of global states of G.

The Control Problem presentation. The considered product systems are particular in the sense that according to Definition 5, there is *no interaction between the various sub-systems* of G. In this paper, a state-based approach is proposed. In this framework, the purpose of a supervisor will be to coordinate the evolution between each sub-system in a way that they do no evolve into a set of illegal global states of G. In practice, this set that can be locally decomposed according to each sub-system. Separately, they do not correspond to a dangerous state. It is only when all sub-systems are simultaneously in these particular states that the system is itself in a dangerous situation that has to be avoided.

In more formal terms, given a product systems $G = G_1 \parallel \cdots \parallel G_n$, our aim is to solve the SACP for a set of forbidden global states E. This has to be done locally on each component of G. Hence, we first need to decompose this set according to the structure of G. In fact, any set of global states E can be represented as a union of Cartesian products of local sets, i.e.

$$E = \bigcup_{1 \leq j \leq m} E^j, \tag{6}$$

where $\forall 1 \leq j \leq m,\ E^j = E_1^j \times \cdots \times E_n^j$ and $\forall 1 \leq i \leq n,\ E_i^j \subseteq \mathcal{X}_i$ of G_i.
In the sequel, E^j will be called a product set. This decomposition in terms of product sets will be the basis for the expression of global sets that will have to be forbidden by control.

Example 2 *To illustrate this aspect, let us consider the well known Cat & Mouse example [17]. The cat and the mouse movements are respectively modeled by the FSMs* CAT *and* MOUSE *for which the states are respectively C_i and M_i, for $i = 0 \cdots 4$ corresponding to the room in which the animals are (the events will be used to model the movements of the animal from one room to another). The goal of supervisor is to avoid the cat and the mouse to be at the same time in the same place. Following (6), the set of forbidden global states can be decomposed in 5 product sets $E_i = \langle C_i, M_i \rangle$, each one modeling the fact that the cat and the mouse are in the same room.*

Remark 2 *In [15], a more restrictive state-based approach was considered. The control objective was assumed to be separable, i.e. the set of forbidden global states was given by $E = \bigcup_{1 \leq j \leq n} E^j$, where $\forall 1 \leq j \leq n,\ E^j = \mathcal{X}_1 \times \cdots \times \mathcal{X}_{i-1} \times E_j \times \mathcal{X}_{i+1} \times \cdots \times \mathcal{X}_n$. Roughly, the control was to avoid the each sub-system G_j to enter local state of E_j whatever the position of the other subsystems.*

In the next sections, we provide a methodology that locally solves the control problem (i.e. on each component of G without computing the whole system) but produces a global supervisor ensuring the avoidance of E in G.

3.2 The State Avoidance Control Problem (SACP)

Let us now consider a modular system $G = G_1 \parallel \cdots \parallel G_n$ (we recall that $\Sigma_i \cap \Sigma_j = \emptyset$). Our aim is to solve the state avoidance control problem for a set of forbidden global states of the form $E = \cup_{1 \leq j \leq m} E^j$, as described in Section 3.1. We first focus on the case where *the set of forbidden global states E is reduced to a product set* of the form $E_1 \times \cdots \times E_n$ (with $E_i \subseteq \mathcal{X}_i$ of G_i), i.e. all the global states of G that belongs to the set $\{\langle x_1, \ldots, x_n \rangle \in \mathcal{X}^F \mid \forall i \leq n,\ x_i \in E_i\}$ have to be forbidden by control.

In a first step, we consider a system modeled by two FSMs and then generalize the results to an arbitrary number of FSMs. Assume given two FSMs $G_i = \langle \Sigma_i, \mathcal{X}_i, \mathcal{X}_{o_i}, \mathcal{X}_{f_i}, \delta_i \rangle$, i=1,2, and two set of local states $E_1 \subseteq \mathcal{X}_1$ and $E_2 \subseteq \mathcal{X}_2$. The control objective, we are interested in, consists in avoiding global states of $E_1 \times E_2$ (i.e. of the form $\langle e_1, e_2 \rangle$, with $e_i \in E_i,\ i = 1, 2$), to be reachable in $G_1 \parallel G_2$.
Based on Definition 4, we first compute $\mathcal{I}_i(E_i)$ and $\mathcal{F}_i(E_i)$[4], i=1,2 and we denote by $\mathcal{S}_1 = (S_1, \mathcal{X}'_{o_1})$ (resp. by $\mathcal{S}_2 = (S_2, \mathcal{X}'_{o_2})$) the maximal supervisor that avoids states in E_1 (resp. E_2) to be reachable in G_1 (resp. G_2). These supervisors are computed according to Prop. 1. Note that some supervisors $\mathcal{S}_i$ ensuring the avoidance of E_i in G_i may be trivial and/or blocking.
At this point, one can see that $G' = (\mathcal{S}_1/G_1) \parallel (\mathcal{S}_2/G_2)$ would solve the problem (i.e. global states that belong to $E_1 \times E_2$ are not reachable in G') but would not be maximal (i.e. the control policy is too restrictive). This basically comes from the fact that this control objective is not separable (according to the definition of [21]). However, the next proposition shows that by combining the informations of both supervisors, it is possible to obtain a supervisor that is maximal.

Proposition 3 *With the preceding notations, let $\mathcal{S}_1$ (resp. $\mathcal{S}_2$) be the maximal supervisor that avoids states in E_1 (resp. E_2) to be reachable in G_1 (resp. G_2). Consider now the supervisor $\mathcal{S} = (S, \mathcal{X}'_o)$ defined by*

$$\begin{cases} \forall \langle x_1, x_2 \rangle \in \mathcal{X},\ S(\langle x_1, x_2 \rangle) = \begin{cases} S_1(x_1) \text{ if } x_2 \in \mathcal{I}_2(E_2) \text{ and } x_1 \in \mathcal{F}_1(E_1) \\ S_2(x_2) \text{ if } x_1 \in \mathcal{I}_1(E_1) \text{ and } x_2 \in \mathcal{F}_2(E_2) \\ \emptyset \text{ Otherwise} \end{cases} \\ \mathcal{X}'_o = (\mathcal{X}'_{o_1} \times \mathcal{X}_{o_2}) \cup (\mathcal{X}_{o_1} \times \mathcal{X}'_{o_2}) \end{cases}$$

ensures the avoidance of $E_1 \times E_2$ in $G_1 \parallel G_2$ and is maximal. Moreover, If either $\mathcal{S}_1$ or $\mathcal{S}_2$ is not trivial, then $\mathcal{S}$ is not trivial.

According to the global state $\langle x_1, x_2 \rangle$ that has been reached so far under the control of $\mathcal{S}$, the global supervisor takes its control decision according to the two local supervisors. Hence, if $x_2 \in \mathcal{I}_2(E_2)$ and $x_1 \in \mathcal{F}_1(E_1)$, it means that if the supervisor allows an event $\sigma \in \Sigma_c$ such that $\delta_1(\sigma, x_1) \in \mathcal{I}_1(E_1)$, then G will enter a weak forbidden global state (i.e. that belongs to $\mathcal{I}_1(E_1) \times \mathcal{I}_2(E_2)$). Hence the supervisor $\mathcal{S}_1$ becomes active. Reciprocally, if $x_1 \in \mathcal{I}_1(E_1)$ and $x_2 \in \mathcal{F}_2(E_2)$, the supervisor $\mathcal{S}_2$ is activated. In all the other cases, there is no need for the supervisor $\mathcal{S}$ to disable events as there still exists a way to control G in order to avoid the system to enter a weak forbidden global state.
In order to prove Proposition 3, we first need the following lemma:

Lemma 1 *With the notations of proposition 3, there exists a non-trivial supervisor ensuring the non-reachability of $E_1 \times E_2$ in $G_1 \parallel G_2$, if and only if either $\mathcal{S}_1$ or $\mathcal{S}_2$ is not trivial.*

Proof : If both are trivial, it means that $\forall x_{o_1} \in \mathcal{X}_{o_1}$ (resp. $\forall x_{o_2} \in \mathcal{X}_{o_2}$) there exists in G_1 (resp. in G_2) a sequence $s_1 \in \Sigma^*_{uc_1}$, s.t. $\delta_1(s_1, x_{o_1}) \in E_1$ (resp. $s_2 \in \Sigma^*_{uc_2}$, s.t. $\delta_2(s_2, x_{o_2}) \in E_2$). Hence the sequence, $s_1 s_2 \in (\Sigma_{uc_1} \cup \Sigma_{uc_2})^*$ constitutes an uncontrollable sequence in $G_1 \parallel G_2$ that, starting in $\langle x_{o_1}, x_{o_2} \rangle$, makes evolve the plant into a state of $E_1 \times E_2$ (Note that this sequence is actually admissible in $G_1 \parallel G_2$ as $\Sigma_1 \cap \Sigma_2 = \emptyset$). It entails that there is no non-trivial supervisor for $G_1 \parallel G_2$.

Suppose now that $\mathcal{S}_1$ is not a trivial supervisor and consider the FSM $H = (\mathcal{S}_1/G_1) \parallel G_2$. By definition of $\mathcal{S}_1$, $\forall e_1 \in E_1$, $\forall e_2 \in E_2$, the global state $\langle e_1, e_2 \rangle$ is not reachable in H (since e_1 is not in $\mathcal{S}_1/G_1$), which entails that states of the form $E_1 \times E_2$ are not reachable in H. Moreover H is controllable w.r.t. $G_1 \parallel G_2$, since both S_1/G_1 and G_2 are [15]. So, there exists a non-trivial supervisor for $G_1 \parallel G_2$. $\diamond$

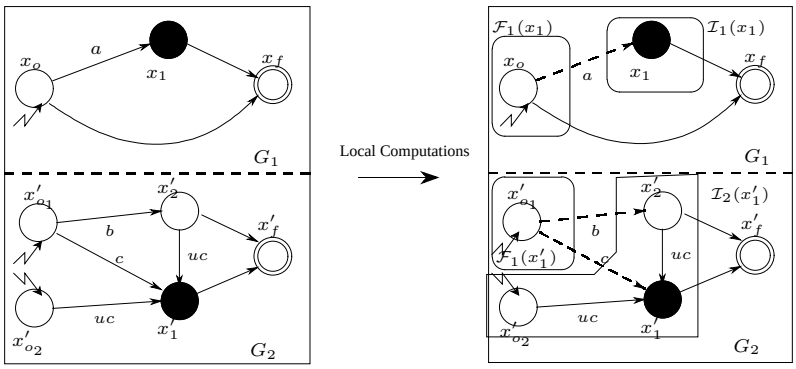
Let us now come back to the proof of Proposition 3.

[4] $\mathcal{I}_i$ and $\mathcal{F}_i$ are computed according to (1) and (2) w.r.t. G_i and E_i.

Proof [Prop. 3]: According to lemma 1, we can suppose that either $\mathcal{S}_1$ or $\mathcal{S}_2$ is non-trivial. Now, if $\langle x_{o_1}, x_{o_2}\rangle \notin \mathcal{X}'_o$, then it means that $x_{o_1} \in \mathcal{I}_1(E_1)$ and $x_{o_2} \in \mathcal{I}_2(E_2)$, which entails that there exists a state $\langle e_1, e_2\rangle \in E_1 \times E_2$, which is reachable via from $\langle x_{o_1}, x_{o_2}\rangle$ an uncontrollable trajectory. Hence, these initial states do not have to be considered. Assume now that $\langle x_{o_1}, x_{o_2}\rangle \in \mathcal{X}'_o$, then it is clear that the action of S prevents $E_1 \times E_2$ from being reachable as whenever G enters a global state $\langle x_1, x_2\rangle$, such that $x_1 \in \mathcal{I}_1(E_1)$ and $x_2 \in \mathcal{F}_2(E_2)$ (resp. $x_1 \in \mathcal{F}_1(E_1)$ and $x_2 \in \mathcal{I}_2(E_2)$), then the local supervisor $\mathcal{S}_2$ (resp. $\mathcal{S}_1$) becomes active and prevent G to enter a state that belongs to $\mathcal{I}_1(E_1) \times \mathcal{I}_2(E_2)$.

Assume now that $\mathcal{S}$ is not maximal and let us call $\mathcal{S}'$ a maximal supervisor S. It means that there exists in $G_1 \parallel G_2$ at least one global state $\langle x_1, x_2\rangle \in \mathcal{X}_1 \times \mathcal{X}_2$ reachable under the control of $\mathcal{S}$ and $\sigma \in (\Sigma_{c_1} \cup \Sigma_{c_2})$, s.t. $\delta_{S'}(\sigma, \langle x_1, x_2\rangle)!$[5] and this transition is not allowed under the control of $\mathcal{S}$. Assume $\sigma \in \Sigma_{c_1}$. Now, according to the structure of S, it means that $x_2 \in \mathcal{I}_2(E_2),\ x_1 \in \mathcal{F}_1(E_1)$ and $\delta_1(\sigma, x_1) \in \mathcal{I}_1(E_1)$. It entails that $\langle \delta_1(\sigma, x_1), x_2\rangle \in \mathcal{I}_1(E_1) \times \mathcal{I}_2(E_2)$ can be reached via σ under the control of $\mathcal{S}'$, which is not admissible. ◇

Example 3 *Consider the two following FSMs G_1 and G_2, the objective is to avoid in $G_1 \parallel G_2$ the global state $x = \langle x_1, x'_1\rangle$ to be reachable. We first compute the maximal supervisors $\mathcal{S}_1$ (resp. $\mathcal{S}_2$) ensuring the avoidance of x_1 in G_1 (resp. x'_1 in G_2). The action of the supervisors is represented by dash arrows in the Figure 2(b). Now, following Proposition 3, the supervisor $\mathcal{S} = (S_x, \mathcal{X}_{o_{\{x\}}})$ is given by :*



(a): $G = G_1 \parallel G_2$ (b): The resulting controlled system

Figure 2: A simple Example

$\mathcal{X}_{o_{\{x\}}} = \{\langle x_o, x'_{o_1}\rangle, \langle x_o, x'_{o_2}\rangle\}$ *and* $S_x(\langle x_o, x'_{o_2}\rangle) = S_x(\langle x_o, x'_1\rangle) = S_x(\langle x_o, x'_2\rangle) = \{a\}$ *because G_1 is in $x_o \in \mathcal{F}_1(x_1)$ and G_2 is in $\mathcal{I}_2(x'_1)$. Hence, $\mathcal{S}_1$ becomes active.* $S(\langle x_1, x'_{o_1}\rangle) = \{b, c\}$ *(G_1 is in $\mathcal{I}_1(x_1)$ and G_2 in $\mathcal{F}_2(x'_1)$. Therefore $\mathcal{S}_2$ is active). For all the other states, $\mathcal{S}$ is not active.* ◇

Next, we extend the results to an arbitrary number of FSMs.

Proposition 4 *Assume given n FSMs G_i of the form $\langle \Sigma_i, \mathcal{X}_i, \mathcal{X}_{o_i}, \mathcal{X}_{f_i}, \delta_i\rangle,\ i = 1, \ldots, n$, with $\Sigma_i \cap \Sigma_j = \emptyset, i \neq j$ and n subsets of states $(E_i)_{i \leq n}$. Consider the sets $\mathcal{F}_i(E_i),\ \mathcal{I}_i(E_i)$ as defined in Definition 4 as well as the corresponding maximal supervisors $\mathcal{S}_i$ (possibly trivial).*

[5] i.e. σ is admissible in $\langle x_1, x_2\rangle$ in $\mathcal{S}'/G$.

Let $\mathcal{S}_E = (S_E, \mathcal{X}_{o_E})$ be such that $\forall x = \langle x_1, \ldots, x_n \rangle$,

$$S_E(x) = \begin{cases} S_1(x_1) \text{ if } x_1 \in \mathcal{F}_1(E_1) \text{ and } \forall j \neq 1,\ x_j \in \mathcal{I}_j(E_j) \\ \ldots \\ S_i(x_i) \text{ if } x_i \in \mathcal{F}_i(E_i) \text{ and } \forall j \neq i,\ x_j \in \mathcal{I}_j(E_j) \\ \ldots \\ S_n(x_n) \text{ if } x_n \in \mathcal{F}_n(E_n) \text{ and } \forall j \neq n \in \mathcal{I}_j(E_j) \\ \\ \emptyset \text{ Otherwise} \end{cases}$$

$$\mathcal{X}_{o_E} = \{\langle x_{o_1}, \ldots, x_{o_n} \rangle \in \Pi_{i \leq n} \mathcal{X}_{o_i} \ / \ \exists i,\ x_{o_i} \in \mathcal{X}'_{o_i}\}$$

The supervisor $\mathcal{S}_E$ ensures the avoidance of $E_1 \times \ldots \times E_n$ in $G = G_1 \parallel \cdots \parallel G_n$ and is maximal. Moreover, if there exists at least one supervisor $\mathcal{S}_i$ that is not trivial, then $\mathcal{S}$ is not trivial (i.e. the behavior of controlled plant is not empty). $\diamond$

The proof of this proposition is similar to the one of Proposition 3.
let S_E be a supervisor as in Proposition 4, one can see that at most one supervisor is active at a time. It is the one for which the sub-plant has evolved in its border set of states when the other sub-plants are in a weak forbidden state. In particular this entails that whenever the system under control $\mathcal{S}_E/G$ has reached a global state $x = \langle x_1, \ldots, x_i, \ldots, x_n \rangle$, such that $x_i \in \mathcal{I}(E_i)$, the supervisor $\mathcal{S}_i = (S_i, \mathcal{X}'_{o_i})$ is not active. This means that the local supervisors are active only when it is necessary (this is due to the fact that we check whether $x_i \in \mathcal{F}_i(E_i)$ or not before "activating" S_i). This is the reason why the sets of border states $\mathcal{F}_i$ have been kept in the definition of the supervisor $\mathcal{S}_E$. This allows us not to store in memory the control policies for states that would not be reachable under the control of $\mathcal{S}$ (and that for G and for the subsystems G_i).
The interest of such a method is that the supervisor $\mathcal{S}_E$ is locally computed according to the local supervisors $\mathcal{S}_i$. Therefore, this method avoids to build the whole system and the computation of $\mathcal{S}$ on the resulting system. Hence, it reduces the complexity of the algorithm (See the next paragraph) as well as the memory storage of the supervisor. Moreover, the supervisor itself somehow keeps the structure of the plant as it is represented as a collection of local supervisors. Hence, the way $\mathcal{S}_E$ is built may improve the readability and the understanding of the control effect. However, on-line computations are still needed to know whether an event has to be disabled or not, i.e. to be activated, a local supervisor need to have some information coming from the different components of the system. In particular it has to know whether the other components are in a weak forbidden (or border) state or not.

Remark 3 *Another characterization of $\mathcal{S}_E$ is the following. Considering the global plant G and the set of forbidden states $E = E_1 \times \cdots \times E_n$, the weak set of forbidden configuration is given by $\mathcal{I}(E) = \mathcal{I}_1(E_1) \times \cdots \times \mathcal{I}_n(E_n)$, whereas the border set is given by*

$$\mathcal{F}(E) = \bigcup_{1 \leq i \leq n} \mathcal{I}_1(E_1) \times \cdots \times \mathcal{I}_{i-1}(E_{i-1}) \times \mathcal{F}_i(E_i) \times \mathcal{I}_{i+1}(E_{i+1}) \times \cdots \times \mathcal{I}_n(E_n)$$

Further, $\mathcal{S}_E$ can be computed as in Proposition 1. Note that in this case, even if all the computations have been done locally, we obtain a global supervisor that does not reflect the structure of the plant to be controlled (see [8] for more details). $\diamond$

Let us now present the main theorem of this section. It shows hows to avoid a set of global states E (as defined by (6)) to be reachable in a product system.

Theorem 1 *Let $G = G_1 \parallel \cdots \parallel G_n$ be the plant to be controlled and a set $E = \bigcup_{1 \leq j \leq m} E^j$ where $\forall 1 \leq j \leq m,\ E^j = E^j_1 \times \cdots \times E^j_n$ and $E^j_i \subseteq \mathcal{X}_i$ for $1 \leq i \leq n$. Let $\mathcal{S}_{E^j} = (S_{E^j}, \mathcal{X}_{o_{E^j}})$ be the maximal supervisors computed w.r.t. G and E^j, then $\mathcal{S}_E = (S, \mathcal{X}_{o_E})$, where $\forall x = \langle x_1, \ldots, x_n \rangle \in \mathcal{X}$,*

$$\begin{cases} S(x) = S_{E^1}(x) \cup \cdots \cup S_{E^m}(x) \\ \mathcal{X}_{o_E} = \mathcal{X}_{o_{E^1}} \cap \cdots \cap \mathcal{X}_{o_{E^m}} \end{cases} \tag{7}$$

ensures the avoidance E in G and is maximal.

Proof : Based on Prop. 4, $\forall 1 \leq j \leq m, \mathcal{S}_{E^j}$ is a controllable and maximal supervisor with respect to $G,\ \Sigma_{uc}$ and the control objective E^j. The modularity result of Prop. 2 gives the results. $\diamond$

Finally, note that this kind of control objectives cannot be efficiently solved using the method presented in [6, 11] since G has to be built (i.e. the FSM $G_1 \parallel \cdots \parallel G_n$ has to be computed), as the objectives concern the whole system. However, our method can be used in complement to the one of [6, 11] whenever the control objective does not concern the whole objective (i.e our method can be used to compute the controller on the sub-machines for which forbidden interactions are required). The global supervisor can then be inferred using the methods of [6, 11].

The complexity aspect. Let us now discuss about complexity of the control synthesis phase. Given an FSM with N states and a set of states to be avoided by control, assume that the complexity of this controller synthesis phase is in $\mathcal{O}(N|\Sigma|)$. Let us now consider a system G of the form $G_1 \parallel \cdots \parallel G_n$ where each G_j contains N states (we assume that $max_i(|\Sigma_i|) = k$). Due to the asynchronous and the trim assumptions, the number of states of G is N^n. Hence, using classical techniques, the state avoidance control problem is in $\mathcal{O}(n.k.N^n))$. In our case, given a product set $E = E_1 \times \cdots \times E_n$, to be avoided by control, we locally compute the supervisor on each local component of G. Hence the global complexity is in $\mathcal{O}(n.k.N)$. More generally, given a set of forbidden global states E, then this set can be decomposed into a union of product sets. Assume that E is composed of m product sets, then the supervisor computation complexity is in $\mathcal{O}(m.n.k.N)$. However one has also to take into account the computations that have to be done on-line when controlling the plant. Indeed, deciding which supervisor have to be activated given one configuration, is done at execution time. This can be done in $\mathcal{O}(m.n.N)$, which is an acceptable complexity. However, if the objectives are not well structured, the number m of product sets to be forbidden can be important. In this case, the on-line computation may be important as well and one has to find a good set of states representation as Binary Decision Diagrams [3] for example[6].

3.3 Non-Blocking SACP

In the previous section, it may happen that the resulting controlled system be blocking. We now give sufficient conditions for the controlled system obtained using the methodology of Section 3.2 to be non-blocking. Assume given a plant G of the form $G_1 \parallel \cdots \parallel G_n$ and a forbidden product set $E = E_1 \times \cdots \times E_n$, then the next proposition gives a sufficient condition for the controlled system to be non-blocking.

Proposition 5 *Let $\mathcal{S}_E$ be the supervisor computed as in Proposition 4 w.r.t. G and E. Then, if $\forall 1 \leq i \leq n$,*

(a) *either $\forall x_i \in \mathcal{X}_i \setminus \mathcal{I}_i(E_i),\ \exists x_{f_i} \in \mathcal{X}_{f_i} \setminus \mathcal{I}_i(E_i)$ that is reachable from x_i in S_{E_i}/G_i*

(b) *or $\exists j \neq i,\ \forall x_j \in \mathcal{X}_j,\ \exists x_{f_j} \in \mathcal{X}_{f_j} \setminus \mathcal{I}_j(E_j)$ that is reachable from x_j in G_j*

then $\mathcal{S}_E = \mathcal{S}^{\uparrow}$, where $\mathcal{S}^{\uparrow}$ is the most permissive non-blocking supervisor ensuring the avoidance of E in G.

[6] see e.g. [24, 14] for controller synthesis tools based on a BDD implementation.

Proof : Let $1 \leq i \leq n$.
• Assume that i is such that **(a)** is true. Let $x = \langle x_1, \ldots, x_n \rangle$ be a reachable configuration of S_E/G.

(1) If $x_i \notin \mathcal{I}_i(E_i)$, then it exists $x_{f_i} \in \mathcal{X}_{f_i} \setminus \mathcal{I}_i(E_i)$ that is reachable from x_i under the control of S_{E_i} (according to **(a)**). This entails that the state $x' = \langle x_1, \ldots, x_{i-1}, x_{f_i}, x_{i+1}, \ldots, x_m \rangle$ is reachable from x under the control of S_E. Now, since $x_{f_i} \notin \mathcal{I}_i(E_i)$, no control is applied on G_j for $j \neq i$. Therefore for each $j \neq i$, it exists $x_{f_j} \in \mathcal{X}_{f_j}$ such that $\langle x_{f_1}, \ldots, x_{f_m} \rangle$ is reachable from x' in S_E/G.

(2) Assume now that x is such that $x_i \in \mathcal{I}(E_i)$, then it exists $j \neq i$ such that $x_j \notin \mathcal{I}_j(E_j)$ (otherwise, we would be in a forbidden configuration). Now, if j is such that **(a)** holds, we can apply the proof of (1) to show that there exists a reachable final state under the control of S_E. if j is such that **(b)** holds, then $\exists j' \neq j$ such that $\forall x_{j'} \in \mathcal{X}_{j'}$, $\exists x_{f_{j'}} \in \mathcal{X}_{f_{j'}} \setminus \mathcal{I}_{j'}(E_{j'})$ that is reachable from $x_{j'}$ in $G_{j'}$.

Since $x_j \notin \mathcal{I}_j(E_j)$, there is no control applied on $G_{j'}$. Hence, according to **(b)** it exists $x_{f_{j'}} \in \mathcal{X}_{f_{j'}} \setminus \mathcal{I}_{j'}(E_{j'})$ such that $x' = \langle x_1, \ldots, x_{j'-1}, x_{f_{j'}}, x_{j'-1}, \ldots, x_m \rangle$ is reachable from x in S_E/G. And since $x_{f_{j'}} \notin \mathcal{I}_{j'}(E_{j'})$, for each $k \neq j'$, there is no control applied on G_k and it exists $x_{f_k} \in \mathcal{X}_{f_k}$ such that $\langle x_{f_1}, \ldots, x_{f_m} \rangle$ is reachable from x'.

• Assume now that i is such that **(b)** is true. Let $j \neq i$, be such that $\forall x_j \in \mathcal{X}_j$, $\exists x_{f_j} \in \mathcal{X}_{f_j} \setminus \mathcal{I}_j(E_j)$ that is reachable from x_j in G_j.

(1) If $x_j \in \mathcal{I}_j(E_j)$, then there is non control applied on G_j and according to **(b)**, it exists $x_{f_j} \in \mathcal{X}_{f_j} \setminus \mathcal{I}_j(E_j)$ such that $x' = \langle x_1, \ldots, x_{j-1}, x_{f_j}, x_{j+1}, \ldots, x_m \rangle$ is reachable from x in S_E/G. And since $x_{f_j} \notin \mathcal{I}_j(E_j)$, for each $k \neq j$ it exists $x_{f_k} \in \mathcal{X}_{f_k}$ such that $\langle x_{f_1}, \ldots, x_{f_m} \rangle$ is reachable from x'.

(2) If $x_j \notin \mathcal{I}_j(E_j)$, either S_{E_j}/G_j verifies **(a)** and then we have the result (this case was dealt above). Or S_{E_j}/G_j verifies **(b)** and we denote $j' \neq j$ such that $\forall x_{j'} \in \mathcal{X}_{j'}$, $\exists x_{f_{j'}} \in \mathcal{X}_{f_{j'}} \setminus \mathcal{I}_{j'}(E_{j'})$ that is reachable from $x_{j'}$ in $G_{j'}$. In this case, there is no control applied on G_j. Therefore, it exists $x_{f_j} \in \mathcal{X}_{f_j} \setminus \mathcal{I}_j(E_j)$ such that $x' = \langle x_1, \ldots, x_{j-1}, x_{f_j}, x_{j+1}, \ldots, x_m \rangle$ is reachable from x in S_E/G. And since $x_{f_j} \notin \mathcal{I}_j(E_j)$, for each $k \neq j$, $\exists x_{f_k} \in \mathcal{X}_{f_k}$ such that $\langle x_{f_1}, \ldots, x_{f_m} \rangle$ is reachable from x'.

The fact that $\mathcal{S}_E$ is the most permissive supervisor is due to proposition 1. ◇

What the above property states is that the obtained supervisor is blocking whenever there exists a state x_i in an FSM G_i such that the final states of G_i are not reachable from x_i under the control of the local controller $\mathcal{S}_{E_i}$ and for all the other FSMs all the final states belong to the weak forbidden set of states. Obviously, $\mathcal{S}_E$ is blocking only under strong hypothesis. Nevertheless, if these conditions do not occur, then one has to find a way to avoid the blocking configurations. This aspect is currently under investigation.

4 The Hierarchical Finite State Machine

So far, we gave results dealing with the control of asynchronous product of FSMs. We now extend these results to the case of Hierarchical Finite State Machines (HFSM). A Hierarchical Finite State Machines is an FSM which includes new features like the nesting of state machines (inducing the hierarchy) and the parallelism between state machines. From now on, some states (called super-states) of an FSM can be other FSMs. Informally, the meaning of such a hierarchical definition is obtained by recursively substituting each super-state by a set of asynchronous FSMs running in parallel. Such a model is called Hierarchical Finite State Machine (HFSM). We hereby focus on a two-level Finite State Machine, knowing that the results presented in the next section can be extended to a multi-level hierarchical finite state machine (See [15] for a more complete review of the HFSMs).

4.1 Definition of an HFSM

In order to take into account the hierarchy, we need to introduce the notion of structure. It will represent the upper level of the HFSM.

Definition 6 *A structure K is a tuple $\langle \Sigma, \mathcal{X}, \mathcal{B}, \mathcal{X}_o, \mathcal{X}_f, \delta \rangle$, where $\mathcal{X}$ is a set of atomic states, $\mathcal{X}_o \subseteq \mathcal{X}$ is the set of initial states and $\mathcal{X}_f \subseteq \mathcal{X}$ is the set of final states. $\mathcal{B}$ is the set of super-states of K (with $\mathcal{X} \cap \mathcal{B} = \emptyset$). δ is the partial transition function of K defined over $\Sigma \times \{\mathcal{X} \cup \mathcal{B}\} \rightarrow \{\mathcal{X} \cup \mathcal{B}\}$.* •

In the following we will denote by $K^A = \langle \Sigma, \mathcal{X} \cup \mathcal{B}, \mathcal{X}_o, \mathcal{X}_f, \delta \rangle$ the structure K seen as an FSM (i.e. when the super-states are considered as atomic states). The notion of structure allows us to define the notion of Hierarchical Finite State Machines.

Definition 7 *A Hierarchical Finite State Machine $\mathcal{K}$ is given by a tuple $(\langle K, G_1, \ldots, G_n \rangle, Y, I)$, where K is a structure as defined in Def 6 and $\forall 1 \leq i \leq n$, $G_i = \langle \Sigma_i, \mathcal{X}_i, \mathcal{X}_{o_i}, x_{f_i}, \delta_i \rangle$ is an FSM, and Y, I are two functions that characterize the hierarchy and the composition between the FSMs.*

- *$Y : \mathcal{B} \longrightarrow 2^{\langle G_1, \ldots, G_n \rangle}$ is a function which maps each super-state $b \in \mathcal{B}$ on a set of FSMs G_i, with $i \geq 1$. We use J_b as $\{j \leq n | \ G_j \in Y(b)\}$. The FSMs of $Y(b)$ behave asynchronously (see Def. 5).*
- *I is an Input function that gives access to the set of initial states that are reached when triggering an event that makes the system evolve into a super-state b. $\forall b \in \mathcal{B}$, $I(b)$ is a function defined over $\Pi_{j \in J_b} \mathcal{X}_{o_j} \rightarrow 2^{\Sigma_i}$. Given a super-state $b \in \mathcal{B}$, and $x_o = \langle x_{o_1}, \ldots, x_{o_{\| J_b \|}} \rangle$ a tuple of initial states, $I(b)(x_o)$ corresponds to the events that make the system go from its current state into x_o.* •

K will be called the *top-level* of $\mathcal{K}$. An example of HFSM is given in Figure 3.

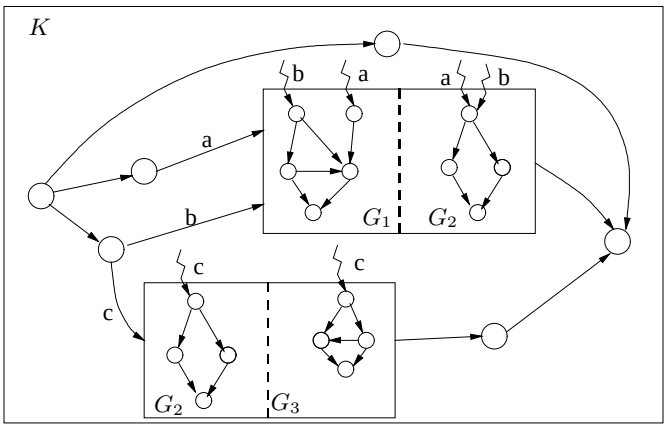


Figure 3: An example of a two-level FSM

Assumptions. In order to be able to perform control on HFSM, we need to make some assumptions on it:

1. $\forall i, j$, s.t. $\exists b \in \mathcal{B},\ G_i,\ G_j \in Y(b),\ \Sigma_i \cap \Sigma_j = \emptyset$ (*asynchrony of parallel FSMs*).
2. Let $b \in \mathcal{B}$ and let $(G_j)_{j \in J_b} = Y(b)$ be the corresponding FSMs attached to b. Then,

$$\delta^{-1}(b) \subseteq \bigcup_{x_o \in \Pi_{j \in J_b}(\mathcal{X}_{o_j})} (I(b)(x_o)) \text{ and } \forall x_o, x'_o \in \Pi_{j \in J_b}(\mathcal{X}_{o_j}), I(b)(x_o) \cap I(b)(x'_o) = \emptyset$$

   i.e. entering the super-state b is deterministic and each event leading to b is taken into account.

The behavior of $\mathcal{K}$. Let $\mathcal{K} = (\langle K, G_1, \ldots, G_n\rangle, Y, I)$ be an HFSM. $\mathcal{K}$ is initialized in one of the initial states of K and as long as no super-state is reached, the behavior of $\mathcal{K}$ corresponds to the one of the FSM K^A. Assume now that the plant is in a state x (at the top-level) such that $\delta(\sigma, x) = b \in \mathcal{B}$ and that σ is triggered. Then all the structures of $Y(b)$ are simultaneously activated and entered in one of their initial states according to $I(b)$, i.e. $\mathcal{K}$ in the configuration $[b, x_o] = [b, \langle x_{o_{j_1}}, \ldots, x_{o_{j_{\|J_b\|}}}\rangle]$, such that $\sigma \in I(b)(x_o)$. Further, the different structures evolve asynchronously following Definition 5. However, in order to evolve out of a super-state b, there is a synchronization between the different structures of $Y(b) = (G_i)_{i\in J_b}$ on their final state (we recall that $\forall i,\ G_i$ has a unique final state). Hence, an event $\sigma \in \delta(b)$ can be triggered in a super-state b whenever each substructure of $Y(b)$ is in its corresponding final state (i.e., there is no preemption); the output of a super-state is synchronized with the end of each of the tasks associated with the different structures involved in this super-state).

Expanded two-level HFSM. Given a HFSM $\mathcal{K} = (\langle K, G_1, \ldots, G_n\rangle, Y, I)$, we can make correspond an FSM. It is obtained as follows. To each super-state $b \in \mathcal{B}$, we associate its corresponding FSM G_b obtained by performing the asynchronous product between each FSM of $Y(b)$, i.e. $G_b = \langle \Sigma_b, \mathcal{X}_b, \mathcal{X}_{o_b}, x_{f_b}, \delta_b\rangle = \|_{j\in J_b} G_j$. To expand the structure K, we replace each super-state b of $\mathcal{B}$ by its corresponding FSM G_b and we connect the initial states of K_b^F to the states of K according to I (resp. for the final state). The result is an FSM, denoted by $\mathcal{K}^F$. Formally, $\mathcal{K}^F = \langle \Sigma^F, \mathcal{X}^F, \mathcal{X}_o^F, \mathcal{X}_f^F, \delta^F\rangle$, where each component is defined by:
• $\Sigma^F = \Sigma \cup \bigcup_{b\in\mathcal{B}}\{\Sigma_b\}$, $\mathcal{X}_o^F = \mathcal{X}_o,\ \mathcal{X}_f^F = \mathcal{X}_f$ and
• $\mathcal{X}^F = \mathcal{X} \cup \{[b, \langle x_1, \ldots, x_{\|J_b\|}\rangle] \mid \forall b \in \mathcal{B}, \forall \langle x_1, \ldots, x_{\|J_b\|}\rangle \in \mathcal{X}_b\}$
• The partial transition function δ^F is defined as follows:

- $\forall x, x' \in \mathcal{X},\ \forall b, b' \in \mathcal{B},\ \forall \sigma \in \Sigma^F$,
  - $\delta^F(\sigma, x) = x'$ if $x \in \mathcal{X}$ and $\delta(\sigma, x) = x'$
  - $\delta^F(\sigma, x) = [b, x_{o_b}]$ if $x \in \mathcal{X}$, $\delta(\sigma, x) = b \in \mathcal{B}$ and $\sigma \in I(b)(x_{o_b})$.
  - $\delta^F(\sigma, [b, x_{f_b}]) = x$ if $b \in \mathcal{B}$ and $\delta(\sigma, b) = x$, where x_{f_b} is the final state of G_b.
  - $\delta^F(\sigma, [b, x_{f_b}]) = [b', x_{o_{b'}}]$ if $\delta(\sigma, b) = b'$, $\sigma \in I(b')(x_{o_{b'}})$ and x_{f_b} is the final state of G_b.
- $\forall b \in \mathcal{B}, \forall x_b \in \mathcal{X}_b, \forall \sigma \in \Sigma_b\ \delta^F(\sigma, [b, x_b]) = [b, \delta_b(\sigma, x_b)]$.

Such an FSM is called the *expanded FSM* of $\mathcal{K}$.

4.2 The SACP

In this section, we consider the configuration avoidance problem, namely how to avoid the system to reach some configurations during its evolution. By configuration, we mean a particular state of $\mathcal{K}^F$.

Forbidden configurations : let $\mathcal{K} = (\langle K, G_1, \ldots, G_n\rangle, Y, I)$ be an HFSM, with $K = (\Sigma, \mathcal{X}, \mathcal{B}, \mathcal{X}_o, \mathcal{X}_f, \delta)$. Given $b \in \mathcal{B}$, we denote by E^b the union of product sets $E^b = \bigcup_{1\le j\le m_b} E^{b,j}$ where $E^{b,j}$ is a product set of the form $E^{b,j} = E_{j_1}^{b,j} \times \cdots \times E_{j_{\|J_b\|}}^{b,j}$ et $E_{j_i}^{b,j} \subseteq \mathcal{X}_{j_i}$ pour $j_i \in J_b$.
For simplicity, the set of configurations $[b, \langle x_{j_1}, \ldots, x_{j_{\|J_b\|}}\rangle]$ such that $\langle x_{j_1}, \ldots, x_{j_{\|J_b\|}}\rangle \in E^b$ is denoted $[b, E^b]$. Now, every set of configurations of $\mathcal{K}^F$ can be represented by a set E of the form

$$E = E_0 \bigcup \left(\bigcup_{b\in\mathcal{B}} [b, E^b] \right) \tag{8}$$

where $E_0 \subseteq \mathcal{X}$. This set represents the forbidden configurations at the top-level of $\mathcal{K}$, whereas $[b, E^b]$ corresponds to the forbidden configurations at the lower level (i.e. inside the super-state b). As in Section 3, the idea of the control is to compute supervisors separately for each structure/FSMs (without expanding the system), and then to build a global supervisor.

Control of a structure First, the definition of weak forbidden set of states introduced in Definition 4 need to be extended in order to take into account the super-states. The idea is that we do not want to remove a super-state by control as there possibly exists a way to control the system inside this super-state. *A contrario*, let $b \in \mathcal{B}$ a super-state of K, given a control objective, it may happen that we need to restrict the entering in a super-state. Hence, for $A \subseteq \delta^{-1}(b)$, we introduce $b_{|A}$ the "controlled super-state" b considering it is only reachable by triggering an event of A and we denote by $\mathcal{B}_{|\mathcal{A}} = \{b_{|A} \mid b \in \mathcal{B} \text{ and } A \subseteq \delta^{-1}(b)\}$, the corresponding set of controlled super-states. This kind of states are introduced in order to partially forbid some super-states. Indeed, one can only want some super-states b to be reachable with respect to a subset of $I(b)(.)$. In fact, this is a way to avoid some initial states of b to be reachable at the lower level of the hierarchy.
Based on these remarks and definitions, we extend the definition of weak forbidden set of states introduced in Definition 4:

Definition 8 *Let $K = \langle \Sigma, \mathcal{X}, \mathcal{B}, \mathcal{X}_o, \mathcal{X}_f, \delta \rangle$ be the top-level of an HFSM $\mathcal{K}$ and $e \in \mathcal{X} \cup \mathcal{B}_{|\mathcal{A}}$.*
• If $e \in \mathcal{X}$, then

$$\mathcal{I}_{\mathcal{X}}(e) = \{x \in \mathcal{X} \mid \exists s \in \Sigma_{uc}^*,\ \delta(s,x) = e \text{ and } \forall s' \leq s,\ \delta(s',x) \notin \mathcal{B}\}.$$

$$\mathcal{I}_B(e) = \{b \in \mathcal{B} \mid \exists \sigma \in \Sigma_{uc},\ \delta(\sigma, b) \in \mathcal{I}_{\mathcal{X}}(e)\}$$

• If $e = b_{|A} \in \mathcal{B}_{|\mathcal{A}}$, then

$$\mathcal{I}_{\mathcal{X}}(b_{|A}) = \{x \in \mathcal{X} \mid \exists s \in \Sigma_{uc}^*,\ \exists \sigma \in \Sigma_{uc} \cap A, \delta(s\sigma, x) = b \text{ and } \forall s' \leq s,\ \delta(s',x) \notin \mathcal{B}\}$$

$$\mathcal{I}_B(b_{|A}) = \{b' \in \mathcal{B} \mid \exists \sigma \in \Sigma_{uc},\ (\delta(\sigma, b') \in \mathcal{I}_{\mathcal{X}}(b_{|A})) \text{ or } (\sigma \in \Sigma_{uc} \cap A \text{ and } \delta(\sigma, b') = b)\}$$

Finally, given $E \subseteq \mathcal{X} \cup \mathcal{B}_{|\mathcal{A}}$, $\mathcal{I}_{\mathcal{X}}(E) = \cup_{e \in E} \mathcal{I}_{\mathcal{X}}(e)$[7] and $\mathcal{I}_B(E) = \cup_{e \in E} \mathcal{I}_B(e)$

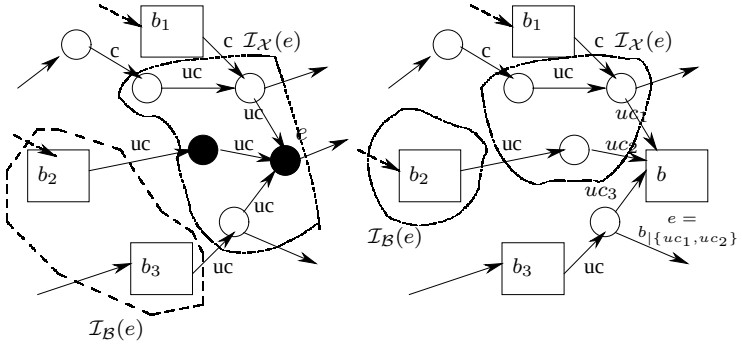


Figure 4: $\mathcal{I}_{\mathcal{X}}(e)$, $\mathcal{I}_{\mathcal{B}}(e)$ through an example

[7] Note that if $\mathcal{B} = \emptyset$, then $\mathcal{I}_{\mathcal{X}}(E) = \mathcal{I}(E)$ where $\mathcal{I}(E)$ is computed as in (1).

Intuitively speaking, given $e \in E$, if $e \in \mathcal{X}$, $\mathcal{I}_{\mathcal{X}}(e)$ (resp. $\mathcal{I}_B(e)$) represents the set of atomic states (resp. super-states) of K from which e can be reached via an uncontrollable trajectory that only traverses atomic states. if $e = b_{|A} \in \mathcal{B}_{|\mathcal{A}}$, the meaning of $\mathcal{I}_{\mathcal{X}}(e)$ and $\mathcal{I}_B(e)$ is similar except that we ask the last event of the uncontrollable trajectories to belong to A.
The next operator Φ will be useful to compute the set of weak forbidden configurations by going-up/down in the hierarchy of the plant. Indeed, if an initial state of a super-state has to be forbidden by control, then at the top-level, a supervisor has to avoid the system to enter the super-state via this initial state. Conversely, the final state of a super-state that may lead to a forbidden configuration via an uncontrollable trajectory has also to be forbidden by control. This is captured by the definition 9, but first we intuitively explain Φ through an example:

Example 4 *Assume that the configuration $[b_2, \langle x_9, x_{10}\rangle]$ has to be forbidden by control. First it is easy to*

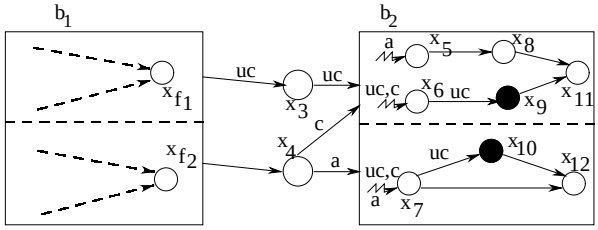


Figure 5: An intuitive example

show that the following set of configurations has to be forbidden: $\{x_6, x_9\} \times \{x_7, x_{10}\}$. It corresponds to the local computations of the weak forbidden set of states. According to this set of forbidden states, one can see that the super-state b_2 must not be reached via the events that lead to $[b_2, \langle x_6, x_7\rangle]$. According to I, b_2 must not be reached via the events c or uc. Thus, $b_{2|\{c,uc\}}$ has to be forbidden (Pt 2 Def. 9). Moreover, as the event uc is uncontrollable, we obtain $\mathcal{I}_{\mathcal{X}}(b_{2|\{c,uc\}}) = \{x_3\}$ and $\mathcal{I}_B(b_{2|\{c,uc\}}) = \{b_1\}$. This entails that x_3 and $[b_1, \langle x_{f_1}, x_{f_2}\rangle]$ are forbidden configurations (Pt 1 Def. 9). Hence, given a forbidden configuration, the Φ function allows to find the configurations or elements of $\mathcal{B}_{|\mathcal{A}}$ that have to be forbidden at the top (resp. lower) level in order to ensure the local control objectives.

Definition 9 *Let $e \in \mathcal{X}^F \cup \mathcal{B}_{|\mathcal{A}}$, , we denote by $\mathcal{X}_{o_b}$ (resp. x_{f_b}) the set of initial states (resp. the final state) of the FSM associated to b (i.e. $K_b = \|_{G_i \in Y(b)} G_i$). Now, we define $\Phi(e)$ as follows:*

1. *If $e \in \mathcal{X} \cup \mathcal{B}_{|\mathcal{A}}$, then $\Phi(e) = \mathcal{I}_{\mathcal{X}}(e) \cup \{[b, x_{f_b}] \mid b \in \mathcal{I}_{\mathcal{B}}(e)\}$*
2. *If $e = [b, \langle x_{j_1}, \ldots, x_{j_{\|J_b\|}}\rangle]$, then given the set $\mathcal{I} = \mathcal{I}_{j_1}(x_1) \times \cdots \times \mathcal{I}_{j_{\|J_b\|}}(x_{j_{\|J_b\|}})$[8],*

$$\Phi(e) = [b, \mathcal{I}] \cup b_{|\bigcup_{x_{o_b} \in \mathcal{I} \cap \mathcal{X}_{ob}} \{I(b)(x_{o_b})\}}$$

Now, given a set $E \subseteq \mathcal{X}^F \cup \mathcal{B}_{|\mathcal{A}}$, $\Phi(E) = \bigcup_{e \in E} \Phi(e)$.

Given $e \in \mathcal{X}^F \cup \mathcal{B}_{|\mathcal{A}}$, if $e \in \mathcal{X} \cup \mathcal{B}_{|\mathcal{A}}$, then $\Phi(e)$ corresponds to the set of weak forbidden configurations, to which we add the final states of the super-states that can lead into e via an uncontrollable trajectory. This

[8] where $\mathcal{I}_{j_i}(x_i)$ corresponds to the weak forbidden set of states of G_i w.r.t. x_i.

way we are going down in the hierarchy (point *1*). Now, if e is a configuration of the form $[b, \langle x_{j_1}, \ldots, x_{j_b}\rangle]$, then $\Phi(e)$ corresponds to the set of weak forbidden configurations inside the super-state b, as well as the restricted super-state itself if some initial states belong to the weak forbidden set of states. Remark that in point *2.* the computations are made locally on each G_i that are involved in the super-states b.

The Supervisor computation

Assume given a set of forbidden configurations of the form $E = E_0 \bigcup \left(\bigcup_{b\in\mathcal{B}}[b, E^b]\right)$, as defined by Equation 8, then the set of weak forbidden configurations is computed by the following fix-point iteration:

$$\mathcal{E}_o = E \text{ and } \mathcal{E}_{i+1} = \Phi(\mathcal{E}_i) \tag{9}$$

Let us call $\mathcal{I}_H(E)$ the result of the previous fix-point computation (it always terminates since the number of states is finite and $\mathcal{E}_i \subseteq \mathcal{E}_{i+1}$). Now, one can see that the set of forbidden configurations can be reorganized as follows:

$$\mathcal{I}_H(E) = \mathcal{X}' \cup \mathcal{B}'_{|\mathcal{A}} \cup \bigcup_{b\in\mathcal{B}} [b, E'^b], \tag{10}$$

where $E'^b = \bigcup_i E'^{b,i}_{j_1} \times \cdots \times E'^{b,i}_{j_{\|J_b\|}}$ and $\mathcal{X}' \subseteq \mathcal{X}$ and $\mathcal{B}'_{|\mathcal{A}} \subseteq \mathcal{B}_{|\mathcal{A}}$. For the super-states, what the above states, is that it is forbidden to enter these super-states through the events that belong to a set of events A. Moreover, each super-state b can be seen as a collection of asynchronous FSMs for which the set of configurations E'^b has to be forbidden. Note that as E'^b is given by a union of product sets, in order to control the behavior of $\mathcal{K}$, we will use the modular methodology explained in Section 3.2 (Theorem 1). Based on the previous remarks, we then have the following property that makes the link with the expanded HFSM.

Proposition 6 *$\mathcal{I}_H(E) \setminus \mathcal{B}'_{|\mathcal{A}} = \mathcal{I}(E)$, where $\mathcal{I}(E)$ is computed with respect to $\mathcal{K}^F$ as in Definition 4.*

In other words, the set $\mathcal{I}_H(E)$[9] corresponds to the weak forbidden set of states $\mathcal{I}(E)$ of the plant $\mathcal{K}^F$. However, compared to the classical methods, all the computations have been performed locally and not on the global plant.

Lemma 2 *With the notation of Proposition 6 and of (9),*

$$\forall i,\ b_{|A} \in \mathcal{E}_i \ \Leftrightarrow\ \{[b, x_{o_b}] \in \mathcal{X}^F |\ I(b)(x_{o_b}) \cap A \neq \emptyset\} \subseteq \mathcal{E}_i.$$

Proof [Prop 6]: Let us first show that $\mathcal{I}_H(E) \setminus \mathcal{B}'_{|\mathcal{A}} \subseteq \mathcal{I}(E)$. The proof proceeds by induction. We will show that $\forall i, \mathcal{E}_i \setminus \mathcal{B}^i_{|\mathcal{A}} \in I(E)$, where $\mathcal{B}^i_{|\mathcal{A}}$ corresponds to the set of super-states that have been partially forbidden after the i^{th} iteration. At step 0, we obviously have that $E \subseteq \mathcal{I}(E)$. At step i, let $\mathcal{E}_i$ be the result of (9) after i iterations. Assume that $\mathcal{E}_k \setminus \mathcal{B}^k_{|\mathcal{A}} \subseteq \mathcal{I}(E)$, $\forall 0 \leq k \leq i$. Let $\mathcal{E}_{i+1} = \Phi(\mathcal{E}_i)$. We have to show that $\mathcal{E}_{i+1} \setminus \mathcal{B}^{i+1}_{|\mathcal{A}} \in I(E)$. To do so, let $e \in \mathcal{E}_i$.

- If $e \in \mathcal{X}$, then based on Def 9, $\Phi(e) = \mathcal{I}_{\mathcal{X}}(e) \cup \{[b, x_{f_b}] \mid b \in \mathcal{I}_{\mathcal{B}}(e)\} \subseteq \mathcal{E}_{i+1}$, but we also have that $\Phi(e) \subseteq \mathcal{I}(E)$ since $e \in \mathcal{I}(E)$ and $\Phi(e)$ only contains states of $\mathcal{X}^F$ from which it is possible to evolve into e via a trace of uncontrollable events.

[9] to which we remove $\mathcal{B}'_{|\mathcal{A}}$ as this set actually corresponds to the set of initial states that are forbidden in a super-states. Hence, they are taken into account by some $[b, E^b]$. See lemma 2

- If $e = [b, \langle x_{j_1}, \ldots, x_{j_{\|J_b\|}} \rangle]$, then $\Phi(e) = [b, \mathcal{I}] \cup b_{|\bigcup_{x_{o_b} \in \mathcal{I} \cap \mathcal{X}_{ob}} \{I(b)(x_{o_b})\}}$, with $\mathcal{I} = \mathcal{I}_{j_1}(x_1) \times \cdots \times \mathcal{I}_{j_{\|J_b\|}}(x_{j_{\|J_b\|}})$. Hence, we have that $[b, \mathcal{I}] \subseteq \mathcal{E}_{i+1}$ and $[b, \mathcal{I}] \subseteq \mathcal{I}(E)$ (for the latter, see Remark 3).
- Finally, if $e = b_{|A} \in \mathcal{B}^i_{|A}$, then according to Lemma 2, we have that $\{[b, x_{o_b}] \in \mathcal{X}^F |\ I(b)(x_{o_b}) \cap A \neq \emptyset\} \subseteq \mathcal{E}_i$ and then to $\mathcal{I}(E)$ (by assumption). Moreover, according to Def 9 *item 1*, $\Phi(b_{|A})$ contains states of $\mathcal{X}$ that lead to $b_{|A}$ (and at the lower level to the states of $\{[b, x_{o_b}] \in \mathcal{X}^F |\ I(b)(x_{o_b}) \cap A \neq \emptyset\}$) via an uncontrollable trajectory. Thus $\Phi(b_{|A}) \subseteq \mathcal{I}(E)$.

Overall we have that $\mathcal{E}_{i+1} \setminus \mathcal{B}^{i+1}_{|A} \subseteq \mathcal{I}(E)$ and finally by induction, $\mathcal{I}_H(E) \setminus \mathcal{B}'_{|\mathcal{A}} \subseteq \mathcal{I}(E)$.
Let us show that $\mathcal{I}(E) \subseteq \mathcal{I}_H(E) \setminus \mathcal{B}'_{|\mathcal{A}}$. To do so, we consider $x \in \mathcal{I}(E)$. Then there exists a sequence of configurations $x = x_1, x_2, \cdots, x_n$ in $\mathcal{X}^F$ and a sequence of events in Σ^F s.t. $x = x_1,\ \delta^F(\sigma_1, x_1) = x_2, \ldots, \delta^F(\sigma_i, x_i) = x_{i+1}, \ldots, \delta^F(\sigma_{n-1}, x_{n-1}) = x_n \in E$ with $\sigma_i \in \Sigma^F_{uc}$[10]. We will show by induction that $\forall i,\ x_i \in \mathcal{I}_H(E)$. By construction, we have that $x_n \in \mathcal{I}_H(E)$ (as $x_n \in E \subseteq \mathcal{I}_H(E)$. Assume that $x_i, \cdots, x_n$ are in $\mathcal{I}_H(E)$.

- If $x_i \in \mathcal{X}$, then according to the definition of δ^F, either $x_{i-1} \in \mathcal{X}$ or $x_{i-1} = [b, \langle x_{f_{j_1}}, \ldots, x_{f_{j_{\|J_b\|}}} \rangle]$. Then for both cases, according to *item 1.* of Def. 9, we have that $x_{i-1} \in \Phi(x_i) \subseteq \mathcal{I}_H(E)$.
- If $x_i = [b, \langle x_{j_1}, \ldots, x_{j_{\|J_b\|}} \rangle]$ (but $\langle x_{j_1}, \ldots, x_{j_{\|J_b\|}} \rangle$ is not an initial state of b) then according to the definition of δ^F, x_{i-1} is also of the form $[b, \langle x'_{j_1}, \ldots, x'_{j_{\|J_b\|}} \rangle]$ and we have that $\langle x'_{j_1}, \ldots, x'_{j_{\|J_b\|}} \rangle \in \mathcal{I}_{j_1}(x_{j_1}) \times \cdots \times \mathcal{I}_{j_{\|J_b\|}}(x_{j_{\|J_b\|}})$ and finally, according to the *item 2.* of Def. 9, we have that $x_{i-1} \in \Phi(x_i) \subseteq \mathcal{I}_H(E)$.
- Finally, if $x_i = [b, x_{o_b}] = [b, \langle x_{o_{j_1}}, \ldots, x_{o_{j_{\|J_b\|}}} \rangle]$, then either $x_{i-1} \in \mathcal{X}$ or x_{i-1} is of the form $[b', \langle x_{f_{j_1}}, \ldots, x_{f_{j_{\|J_{b'}\|}}} \rangle]$. Now based on the *item 2.* of Def. 9, we have that $b_{I(b)(x_{o_b})} \in \Phi(x_i)$. Now if $x_{i-1} \in \mathcal{X}$, then it means that $\delta(\sigma_{i-1}, x_{i-1}) = b$ in K and we have that $x_{i-1} \in \Phi(b_{I(b)(x_{o_b})}) \subseteq \mathcal{I}_H(E)$ (if $x_{i-1} = [b', \langle x_{f_{j_1}}, \ldots, x_{f_{j_{\|J_{b'}\|}}} \rangle]$ the reasoning is similar).

Overall $x \in \mathcal{I}_H(E)$ and finally $\mathcal{I}(E) \subseteq \mathcal{I}_H(E)$. ◇
Based on the previous decomposition (10) of $\mathcal{I}_H(E)$, a supervisor can be extracted. It is performed as follows:

1. $\forall b \in \mathcal{B}$, we compute the supervisor $\mathcal{S}_b = (S_b, \mathcal{X}_{o_b})$ that avoid the set of product sets E'^b to be reachable in $\|_{j \in J_b} G_j$ using the methods developed in Section 3.2. Note that we only need to compute the borders. If some initial states are forbidden in the super-states, then this is taken into account at the upper level, since in this case, this super-state (restricted to the events that does not lead into these initial states) belongs to $\mathcal{B}'_{|\mathcal{A}}$.
2. For the structure K, we compute $\mathcal{S}_K = (S_K, \mathcal{X}'_o)$ defined by

$$\begin{array}{rcl} S_K(e) & = & \{\sigma \in \Sigma_c | \delta(\sigma, e) \in \mathcal{X}' \vee \delta(\sigma, e) = b \text{ with } b_{|A} \in \mathcal{B}'_{|\mathcal{A}} \wedge \sigma \in A\} \\ \mathcal{X}'_o & = & \mathcal{X}_o \setminus \mathcal{X}' \end{array}$$

   Note that as $\mathcal{X}' \cup \mathcal{B}'_{|\mathcal{A}} = \mathcal{I}(\mathcal{X}' \cup \mathcal{B}'_{|\mathcal{A}})$, we only have to compute the border of this set.

[10] $\Sigma^F_{uc} = \Sigma_{uc} \cup \bigcup_i \Sigma_{i,uc}$, where $\Sigma_{i,uc}$ corresponds to the uncontrollable events of G_i.

Proposition 7 *With the preceding notations, let $\mathcal{S} = \langle \mathcal{S}_K, (\mathcal{S}_b)_{b\in\mathcal{B}} \rangle$, be such that*

$$S_E(e) = \begin{cases} S_K(e) \text{ if } e \in \mathcal{X} \\ S_K(b) \cup S_b(x_{f_b}) \text{ if } e = [b, x_{f_b}] \text{ (i.e. if } e \text{ is the final state of } b) \\ S_b(x_{j_1}, \ldots, x_{j_{\|J_b\|}}) \text{ if } e = [b, \langle x_{j_1}, \ldots, x_{j_{\|J_b\|}} \rangle] \text{ but not final} \\ \emptyset \text{ Otherwise} \end{cases} \quad (11)$$
$$\mathcal{X}_{o_E} = \mathcal{X}'_o$$

Then, $\mathcal{S}$ ensures the avoidance of E and is maximal. ◇

According to (11), the control policy of the supervisor is the following:

- When the current configuration of $\mathcal{K}$ is in $\mathcal{X}$ then the supervisor $\mathcal{S}_K$ is activated. $\mathcal{S}_K$ is also activated when the plant under control enters the final state of a super-state. Note that $\mathcal{S}_K$ also takes into account the fact that some initial configurations of a super-state are possibly forbidden.
- When $\mathcal{K}$ enters a super-state b under the control of $\mathcal{S}_K$ (which entails that the super-state is entered through the allowed events) then the supervisor $\mathcal{S}_b$ becomes active, which means the supervisors of the FSMs involved in b are activated if necessary following the methodology of Section 3.3. It is deactivated whenever the plant under control leaves the super-state b.

To conclude this section, let us remark that as in Section 3, we made the necessary efforts not to expand the HFSM in order to compute the supervisor. In particular, the set of weak forbidden configurations has been computed locally on each submachine and on the upper level of the HFSM.

5 Conclusion & Future Works

In this paper we have considered the control of structured plant modeled as Asynchronous Finite States Machines and Hierarchical Finite State Machines. Based on this model, we proposed a methodology allowing the computation of a supervisor solving the State avoidance control problem. The control objective is given as a collection of forbidden configurations that is decomposed according to the local FSMs. Based on this decomposition, we locally solve the problem with respect to the local FSMs and finally we provide a global supervisor ensuring the global property. As all the computations are done on the sub-plants according to the local specification, there is no need to build the global plant, hence reducing the complexity of the supervisor computation.

In the case of modular Plant, we gave a sufficient condition under which the resulting controlled plant is non-blocking. We are currently looking for an algorithm that will force the plant to be non-blocking while still avoiding the computation of the whole state space. Another point of interest would be to extend the model by adding preemption and synchronizations between the FSMs, which is obviously one of the main limitations of the presented work. The generalization of the control objectives is also under investigation.

References

[1] B. Brandin, R. Malik, and P. Dietrich. Incremental system verification and synthesis of minimally restrictive behaviours. In *Proceedings of the American Control Conference*, pages 4056–4061, Chicago, Illinois, June 2000.

[2] Y. Brave and M. Heimann. Control of discrete event systems modeled as hierarchical state machines. *IEEE Transacations on Automatic Control*, 38(12):1803–1819, December 1993.

[3] R. E. Bryant. Symbolic boolean manipulation with ordered binary-decision diagrams. *ACM computing Surveys*, pages 293–318, September 1992.

[4] P. Caines, V. Gupta, and G. Shen. The hierarchical control of ST-finite-state machines. *Systems and Control Letters*, 32:185–192, 1997.

[5] C. Cassandras and S. Lafortune. *Introduction to Discrete Event Systems*. Kluwer Academic Publishers, 1999.

[6] M.H. deQueiroz and J.E.R. Cury. Modular supervisory control of large scale discrete-event systems. In *Discrete Event Systems: Analysis and Control. Proc. WODES'00*, pages 103–110. Kluwer Academic, 2000.

[7] M.H. deQueiroz and J.E.R. Cury. Synthesis and implementation of local modular supervisory control for a manufacturing cell. In *Proceedings of the 6th International Workshop on Discrete Event Systems*, pages 377–382, October 2002.

[8] B. Gaudin and H Marchand. Supervisory control of structured discrete event systems. Research Report 1569, IRISA, November 2003.

[9] P. Gohari-Moghadam. *A linguistic Framework for controller hierarchical DES*. M.a.s.c. thesis, Dept. of Electl. & Comptr. Engrg., University of Toronto, April 1998.

[10] D. Harel and A. Pnueli. On the development of reactive systems. In *Logics and Models of Concurrent Systems*, volume 13 of *NATO ASI Series*, pages 477–498, New York, 1985.

[11] K. kesson, Flordal, and M. Fabian. Exploiting modularity for synthesis and verification of supervisors. In *Proc. of the IFAC*, barcelona, Spain, July 2002.

[12] R.J. Leduc. *Hierarchical Interface Based Supervisory Control*. PhD thesis, Dept. of Elec. & Comp. Engrg., Univ. of Toronto, 2002.

[13] R.J. Leduc, W.M. Wonham, and M. Lawford. Hierarchical interface-based supervisory control: Parallel case. In *Proc. of the 39th Allerton Conf. on Comm., Contr., and Comp.*, pages 386–395, October 2001.

[14] H. Marchand, P. Bournai, M. Le Borgne, and P. Le Guernic. Synthesis of discrete-event controllers based on the signal environment. *Discrete Event Dynamic System : Theory and Applications*, 10(4):347–368, October 2000.

[15] H. Marchand and B. Gaudin. Supervisory control problems of hierarchical finite state machines. In *41th IEEE Conference on Decision and Control*, Las Vegas, USA, December 2002.

[16] P. J. Ramadge and W. M. Wonham. Modular feedback logic for discrete event systems. *SIAM J. Control Optim.*, 25(5):1202–1218, September 1987.

[17] P. J. Ramadge and W. M. Wonham. The control of discrete event systems. *Proceedings of the IEEE; Special issue on Dynamics of Discrete Event Systems*, 77(1):81–98, 1989.

[18] E. Rutten and H. Marchand. Task-level programming for control systems using discrete control synthesis. Research Report 4389, INRIA, February 2002.

[19] C.R.C. Torrico and J.E.R. Cury. Hierarchical supervisory cotnrol of discrete event systems based on s state aggregation. In *Proc. of the IFAC*, barcelona, Spain, July 2002.

[20] T. Ushio. On controllable predicates and languages in discrete-event systems. In *Proc. of the* 28^{th} *Conference on Decision and Control*, pages 123–124, Tampa, Floride, December 1989.

[21] Y. Willner and M. Heymann. Supervisory control of concurrent discrete-event systems. *International Journal of Control*, 54(5):1143–1169, 1991.

[22] K. C. Wong and W. M. Wonham. Hierarchical control of discrete-event systems. *Discrete Event Dynamic Systems*, 6:241–273, 1996.

[23] W. M. Wonham and P. J. Ramadge. Modular supervisory control of discrete event systems. *Mathematics of Control Signals and Systems*, 1:13–30, 1988.

[24] Z.H. Zhang and W.M. Wonham. Stct: An efficient algorithm for supervisory control design. In *Symposium on Supervisory Control of Discrete Event Systems (SCODES2001)*, Paris (France), July 2001.